Presented at SPIE Int. Symp. on Optical Science, Engineering, and Instrumentation: Mathematical Methods in Geophysical *Imaging III, San Diego, CA, July 95*

Unconditionally Stable Explicit, Method for Massively Parallel Solution of Acoustic Wave Equations

Amir Fijany[1] and Paul C. Messina[2]

[1] *Jet Propulsion Laboratory, California Institute of Technology*
[2] *Center for Advanced Computing Research,*

increase in the computational cost per time step, thus preserving the asympt otic computational cost. We discuss the mathematical foundation of this method by first deriving an unconditionally stable explicit algorithm with a second-order accuracy in both time and space. Wc then discuss the derivation of more accurate algorithms (fourth order accurate in time and fourth and higher order accurate in space). Wc also discuss some techniques for a more efficient implementation of resulting algorithms on massively parallel MIMD architectures. The incorporation of the first and higher order absorbing boundary conditions in the algorithms is also analyzed.

11. The Acoustic Wave Equation

Let us assume that the earth behaves as an acoustic medium and that the variations in density can be ignored. Considering a unit square domain Ω with boundary W, tbc propagation of the cnergy into the earth is then governed by the acoustic wave equation:

$$u_{tt} = c^2(u_{xx} + u_{yy}) + f \qquad x, y \epsilon \Omega \text{ and } 0 < t \leq T \tag{2.1}$$

where $u(x,y,t)$ and $c(z, y)$ represent the wave field and the velocity of the medium, respectively. Also,

$$f(x,y,t) = c^2 s(t)\delta(x - x_s)\delta(y - y_s)$$

where $s(t)$ is a tirne-dependent band-limitcd source located at (x_s, y_s). Equation (2.1) is solved subject to the initial conditions

$$u(x,y,0) = 0 \text{ and } u_t(x,y,0) = 0 \qquad x, y \epsilon \Omega \tag{2.2}$$

and Dirichlct boundary conditions

$$u(x,y,t) = 0 \qquad x, y \epsilon \Omega' \text{ and } 0 \leq t \leq T \tag{2.3}$$

The starting point in our algorithmic development is the transformation of (2.1) into a set of first-order hyperbolic equations. To this end, let

$$q = u_t, \; p = u_x, \text{ and } r = u_y \tag{2.4}$$

Equation (2.1) can now be written as

$$q_t = c^2(p_x + r_y) + f \tag{2.5}$$

From

$$\begin{pmatrix} q_t \\ p_t \\ r_t \end{pmatrix} = \begin{pmatrix} 0 & c^2 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} q_x \\ p_x \\ r_x \end{pmatrix} + \begin{pmatrix} 0 & 0 & c^2 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} q_y \\ p_y \\ r_y \end{pmatrix} \tag{2.7}$$

$$\begin{pmatrix} 0 & 0 & c^2 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

Equation (2.7) can be written aa a first-order hyperbolic equation:

$$Q_t = AQ_x + BQ_y + F \tag{2.8}$$

Let $\bar{\delta}_x = \mathrm{Diag}\{\delta_x, \delta_x, \delta_x\}$ and $\bar{\delta}_y = \mathrm{Diag}\{\delta_y, \delta_y, \delta_y\}$ where δ_x and δ_y are the operators of spatial discretization in x and y directions. The semi-discrete form of (2.8) is then given by

$$Q_t = (A\bar{\delta}_x + B\bar{\delta}_y)Q + F \tag{2.9}$$

The analytical solution of (2,9) is given by

$$Q(t) = e^{(A\bar{\delta}_x + B\bar{\delta}_y)t}Q(0) + \int_0^t e^{(A\bar{\delta}_x + B\bar{\delta}_y)(t-\tau)}F(\tau)\,d\tau$$

Let M and $\Delta t = T/M$ denote the number of time steps and the tirne-step size, respectively. Also, let $Q^{(m)}$ and $F^{(m)}$ denote the solution and the source term at the mth time step, respectively. A marching-in-time (or, time-stepping) procedure for solution of (2.9) is then derived as

$$Q^{(m)} = e^{(A\bar{\delta}_x + B\bar{\delta}_y)\Delta t}Q^{(m-1)} + \frac{1}{2}(F^{(m)} + F^{(m-1)})\Delta t \qquad m = 1 \text{ to } M \tag{2.10}$$

with $Q^{(0)} = 0$. Defining

$$E = (A\bar{\delta}_x + B\bar{\delta}_y) = \begin{pmatrix} 0 & c^2\delta_x & c^2\delta_y \\ \delta_x & 0 & 0 \\ \delta_y & 0 & 0 \end{pmatrix} \tag{2.11}$$

we then have

$$Q^{(m)} = e^{E\Delta t}Q^{(m-1)} + \frac{1}{2}(F^{(m)} + F^{(m-1)})\Delta t \qquad m = 1 \text{ to } M \tag{2.12}$$

Remark 1. Although the transformation of the second. order hyperbolic equations into a set of first-order equations is well known [3], for conventional finite-difference methods it is always more efficient to directly discretize (2.1) in time and space rather than (2.7). This results from the fact that such a transformation increases the dimension of matrices and vectors involved in the computation. However, as shown below, this transformation is the key enabling factor in our algorithmic development.

Remark 2. Having computed the vectors $Q^{(m)}$, the vectors $u^{(m)}$ can be obtained from (2.4) by a simple integration. Thus, in the following we mainly concentrate on the computation of vectors $Q^{(m)}$.

III. A Fast Unconditionally Stable Explicit Method for Solution of the AWE

The computation of matrix exponential has been a topic of extensive research works (see, for example, the references in [4]). Moler and Van Loan's work [4] represents a classical survey of various methods for computing matrix exponential. One of the

A. Approximation Techniques for Computing Matrix Exponential

In our derivation we make usc of the following properties of matrix cxponcntial. The power series definition of exponential of a matrix, say θ, is given by

$$e^{\theta} = I + \theta + \frac{1}{2!}\theta^2 + \frac{1}{3!}\theta^3 + \frac{1}{4!}\theta^4 + \cdots \tag{3.1}$$

from which it follows that if $\theta = V\phi V^{-1}$, i.e., if θ is a similarity transformation of ϕ, then

$$e^{\theta} : Ve^{\phi}V^{-1} \tag{3.2}$$

Now, considcr a splittingof θ as $\theta = \theta_1 + \theta_2$. If θ_1 and θ_2 commute then

$$e^{\theta\Delta t} : e^{\theta_1\Delta t}e^{\theta_2\Delta t}$$

If θ_1 and θ_2 do not commute then first- and seccmd-order approximations of c $^{\theta\Delta t}$ arc given by [6]

$$e^{\theta\Delta t} = e^{\theta_1\Delta t}e^{\theta_2\Delta t} + O((\Delta t)^?) \tag{3.3}$$

and

$$e^{\theta\Delta t} = e^{\frac{1}{2}\theta_2\Delta t}e^{\theta_1\Delta t}e^{\frac{1}{2}\theta_2\Delta t} + O((\Delta t)^3) \tag{3.4}$$

In the following, we concentrate on the second-order temporal approximation given by (3.4). Higher-order approximations will bc discussed in §V.C. Note that, the approximation in (3.4) is not unique. In §VI.C, a different second-order approximation with a greater efficiency for parallel computation is discussed. It should be also mentioned that there arc other methods for approximation of matrix exponential. } or example, there is a class of Pade approximation techniques, with various degree of accuracy, for computing matrix exponential. However, these Pade approxirnations either result in explicit methods with conditional stability or in implicit methods with unconditional stability but demanding large linear system solution.

B. Temporal and Spatial *Discretizations*

From (3.4) a second-order temporal approximation of (2.1?) is obtained as

$$Q^{(m)} = e^{\frac{A}{2}\bar{\delta}_x\Delta t}e^{B\bar{\delta}_y\Delta t}e^{\frac{A}{2}\bar{\delta}_x\Delta t}Q^{(m-1)} + \frac{1}{2}(F^{(m)} + F^{(m-1)})\Delta t \qquad m = 1 \text{ to } M \tag{3.5}$$

For spatial discretization of (3.5), let us superimpose a uniform grid with $\Delta x = \Delta y = \frac{1}{N+1}$ on the spatial domain Ω. Again, wc first consider a second-order accurate spatial discretization (higher order discretizations will be discussed in §V.B). With this grid size, $Q^{(m)}$ and $F^{(m)}$ are $3N^2$ vectors and c is a diagonal matrix given by $c = \text{Diag}\{c_{ij}\}\varepsilon\Re^{N^2\times N^2}$. Also,

$$q^{(m)} = \text{Col}\{q_{ij}^{(m)}\},\ p^{(m)} = \text{Col}\{p_{ij}^{(m)}\},\ r^{(m)} = \text{Col}\{r_{ij}^{(m)}\},\ \text{and } f^{(m)} = \text{Col}\{f_{ij}^{(m)}\}\varepsilon\Re^{N^2} \quad i \text{ and } j = 1 \text{ to } N$$

wherein, for example, $q_{ij}^{(m)}$ represents the approximate solution for point $(i\Delta x, j\Delta y)$ at the mth time step and c_{ij} denotes the velocity at the same point. The above representat ion implies an ordering of the vectors elements first in the direction of x (i) and then in the direction of $y(j)$. With this ordering and using a second-order centered finite-difference approximation, the operators δx and δy arc given by

$$\delta x = \frac{1}{2\Delta x}\bar{S} \text{ and } \delta y = \frac{1}{2\Delta x}P\bar{S}P^t \tag{3.6}$$

wherein t indicates the transpose,

$$\bar{S} = \text{Diag}\{S, S, \ldots\ldots S\}\varepsilon\Re^{N^2\times N^2} \text{ with } S = \text{Tridiag}[1, O, -1]\varepsilon\Re^{N\times N}, \tag{3.7}$$

and $P\epsilon\Re^{N^2\times N^2}$ is a permutation matrix that arises in 2D discrete Fourier transform. Specifically, if the two vectors V and $W\epsilon\Re^{N^2}$ are defined as $V = \mathrm{Col}\{V_{ij}\}$ and W : $\mathrm{Col}\{W_{ij}\}$, for i and j = 1 to N, then $V = PW$ implies that $V_{ij} = W_{ji}$. Also, P-1 = P^t since 1' is a permutation matrix and hence orthogonal. The matrices $\frac{A}{2}\bar{\delta}_x\Delta t$ and $B\bar{\delta}_y\Delta t$ in (3.5) can be written as

$$\frac{A}{2}\bar{\delta}_x\Delta t = \alpha \begin{pmatrix} 0 & c^2\bar{S} & 0 \\ \bar{S} & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \text{ and } B\bar{\delta}_y\Delta t = 2\alpha \begin{pmatrix} 0 & 0 & c^2P\bar{S}P^t \\ 0 & 0 & 0 \\ P\bar{S}P^t & 0 & 0 \end{pmatrix} \tag{3.8}$$

where $\alpha = \frac{\Delta t}{4\Delta x}$. From the property of matrix P it follows that

$$c^2P\bar{S}P^t = PP^tc^2P\bar{S}P^t = Pc'^2\bar{S}P^t$$

where c' is obtained from c through the permutation $c' = PcP^t$ which implies that $c'_{ji} = c_{ij}$, for i and $j = 1$ to N. let us define a permutation matrix $\bar{P}_{21} = \mathrm{Diag}\{P,\ I,\ P\}\epsilon\Re^{3N^2\times 3N^2}$ with $\bar{P}_{21}^{-1} = \bar{P}_{21}^t$. The matrices $\frac{A}{2}\bar{\delta}_x\Delta t$ and $B\bar{\delta}_y\Delta t$ can now be written as

$$\frac{A}{2}\bar{\delta}_x\Delta t = \alpha E_1 \text{ and } B\bar{\delta}_y\Delta t = 2\alpha\bar{P}_{21}E_2\bar{P}_{21}^t \tag{3.9}$$

where

$$E_1 = \begin{pmatrix} 0 & c^2\bar{S} & 0 \\ \bar{S} & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \text{ and } E_2 = \begin{pmatrix} 0 & 0 & c'^2\bar{S} \\ 0 & 0 & 0 \\ \bar{S} & 0 & 0 \end{pmatrix} \tag{3.10}$$

Our subsequent derivation can be further simplified by noting that the matrix E_2 can be reduced to a form similar to E_1. To this end, consider a permutation matrix $\bar{P}_{22}\epsilon\Re^{3N^2\times 3N^2}$ given by

$$\bar{P}_{22} = \begin{pmatrix} I & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

It can be then shown that

$$\bar{P}_{22}E_2\bar{P}_{22}^t = E_1' \tag{3.11}$$

where

$$E_1' = \begin{pmatrix} 0 & c'^2\bar{S} & 0 \\ \bar{S} & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Defining another permutation matrix $\bar{P}_2$ given by

$$\bar{P}_2 = \bar{P}_{21}\bar{P}_{22} = \begin{pmatrix} P & 0 & 0 \\ 0 & 0 & I \\ 0 & P & 0 \end{pmatrix} \tag{3.12}$$

it then follows from (3.9) and (3.11) that

$$e^{B\bar{\delta}_y\Delta t} = e^{2\alpha\bar{P}_2E_1'\bar{P}_2^t} = \bar{P}_2e^{2\alpha E_1'}\bar{P}_2^t \tag{3.13}$$

Equation (3.5) can now be written as

$$Q^{(m)} = \mathcal{M}_1\bar{P}_2\mathcal{M}_2\bar{P}_2^t\mathcal{M}_1Q^{(m-1)} + \frac{1}{2}(F^{(m)} + F^{(m-1)})\Delta t \qquad m = 1 \text{ to } M \tag{3.14}$$

where Ml $= e^{\alpha E_1}$ and $\mathcal{M}_2 = e^{2\alpha E_1'}$.

C. A Splitting Technique for *Computing Matrix Exponential*

The key issue in the fast and stable computation of (3.14) is the way by which the matrices M_1 and M_2 are evaluated. To this end, let us first consider the evaluation of matrix M_1. From the structure of matrix F_1 we have

$$\mathcal{M}_1 = \begin{pmatrix} D_1 & 0 \\ 0 & I \end{pmatrix} \tag{3.15}$$

where $D_1 = e^{\alpha G_1}$ and

$$G_1 = \begin{pmatrix} 0 & c^2\bar{S} \\ \bar{S} & 0 \end{pmatrix} \tag{3.16}$$

Therefore, the calculation of the matrix $\mathcal{M}_1$ is reduced to that of matrix D_1 To this end, consider a splitting of matrix S as $S = S_1 + S_2$ where S_1 and $S_2 \varepsilon \Re^{N\times N}$ are lower and upper bidiagonal matrices given by

$$S_1 = \begin{pmatrix} 0 & 0 & 0 & \dots & 0 \\ 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & 1 & 0 \end{pmatrix} \text{ and } S_2 = \begin{pmatrix} 0 & -1 & 0 & \dots & 0 \\ 0 & 0 & -1 & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & -1 \\ 0 & 0 & \dots & 0 & 0 \end{pmatrix} \tag{3.17}$$

Such a splitting of matrix S leads to a corresponding splitting of $\bar{S}$ as $\bar{S} = \bar{S}_1$ -i $\bar{S}_2$ with $\bar{S}_1 = \text{Diag}\{S_1, S_1, \cdots, S_1\}$ and $\bar{S}_2$ =-

$$\alpha G_{11} = P_{11}^t T P_{11}$$

$$T_{ij} = \alpha \begin{pmatrix} 0 & -c_{ij}^2 \\ 1 & 0 \end{pmatrix}$$

where

$$\Theta = \mathrm{Diag}\{\Theta_{11}, \Theta_{12}, \ldots, \Theta_{1N-1}, I, \Theta_{21}, \Theta_{22}, \ldots, \Theta_{2N-1}, I, \ldots \Theta_{N1}, \ldots \ldots \Theta_{NN-1}, I\} \tag{3.24}$$

with $\Theta_{ij} = e^{T_{ij}}$. Note that Θ is also a block diagonal matrix with 2 by 2 submatrices 011 its main diagonal. The computation of matrices Θ_{ij} is discussed in the next section.

A similar procedure can be used for reducing matrix G_{12} to a block diagonal form. Again, consider the two vectors V and W as defined before and a permutation matrix $P_{12} \varepsilon \Re^{2N^2 \times 2N^2}$ wherein $W = P_{12}V$ implies that

$$W_i = \begin{cases} \mathrm{V}_{N^2 + \frac{i+1}{2}} & \text{if } i \text{ is odd} \\ V_{\frac{i}{2}+1} & \text{if } i \text{ is even} \end{cases}$$

with $W_{2N^2} =- V_1$. It can be then shown that

$$\frac{\alpha}{2} G_{12} = P_{12}^t R P_{12} \tag{3.25}$$

where $R \varepsilon \Re^{2N^2 \times 2N^2}$ is a block diagonal matrix with 2 by 2 submatrices on its diagonal as

$$R = \mathrm{Diag}\{R_{12}, R_{13}, \ldots \ldots R_{1N}, \mathrm{O}, R_{22}, R_{23}, \ldots \ldots \mathrm{R2N}, 0, \ldots \ldots R_{N2} \ldots \ldots R_{NN}, 0\} \tag{3.26}$$

with

$$R_{ij} = \frac{\alpha}{2} \begin{pmatrix} 0 & -1 \\ c_{ij}^2 & 0 \end{pmatrix} \tag{3.27}$$

From (3.25) and (3.26), it follows that

$$e^{\alpha G_{12}} = P_{12}^t \Psi P_{12} \tag{3.28}$$

where

$$\Psi = \mathrm{Diag}\{\Psi_{12}, \Psi_{13}, \ldots \ldots \Psi_{1N}, I, \Psi_{22}, \Psi_{23}, \ldots \ldots \Psi_{2N}, I, \ldots, \Psi_{N2} \ldots, \Psi_{NN}, I\} \tag{3.29}$$

with $\Psi_{ij} = ^{R}e$ Thus the matrix Ψ is also block diagonal with 2 by 2 submatrices Ψ_{ij} on its main diagonal. 'l'he computation of matrices Ψ_{ij} is discussed in the section. From (3.19), (3.23), and (3.28) the matrix D_1 is given as a product of a set of sparse matrices

$$D_1 = P_{12}^t \Psi P_{12} P_{11}^t \Theta P_{11} P_{12}^t \Psi P_{12} \tag{3.30}$$

The matrix $\mathcal{M}_2$ can be computed in a similar fashion as that of $\mathcal{M}_1$ by replacing c with c', i.e., c_{ij} with *cji,* and α with 2cr. Briefly, wc have

$$\mathcal{M}_2 = \begin{pmatrix} D_2 & 0 \\ 0 & I \end{pmatrix} \tag{3.31}$$

$$D_2 = e^{2\alpha G_1'} = e^{\alpha G_{12}'} e^{2\alpha G_{11}'} e^{\alpha G_{12}'} = P_{12}^t \Psi' P_{12} P_{11}^t \Theta' P_{11} P_{12}^t \Psi' P_{12} \tag{3.32}$$

where Ψ' and Θ' arc obtained from Ψ and Θ by replacing *cij* with c_{ji} and α with 2α.

IV. Stability and Efficiency of the Method

A. Unconditional *Stability*

For tbc stability analysis of the method we consider the l_2 norm of matrix

$$\mathrm{M} = \mathcal{M}_1 \bar{P}_2 \mathcal{M}_2 \bar{P}_2^t \mathcal{M}_1 \tag{4.1}$$

The unconditional stability of the method follows from the fact that $||\mathcal{M}|| \leq 1$ which is established as follows. To begin, note that the operation $\bar{P}_2 \mathcal{M}_2 \bar{P}_2^t$ represents a similarity transformation of matrix $\mathcal{M}_2$ which preserves its norm. Also, consider the following property of matrix norm as

$$||AB|| \leq ||A||||B|| \tag{4.2}$$

By a successive application of (4.2) it follows that if $||\mathcal{M}_1|| \leq 1$ and $||\mathcal{M}_2|| \leq 1$ then $||\mathcal{M}|| \leq 1$.

First consider matrix $\mathcal{M}_1$. From (3.15) and (3.30) and using the argument based on the similarity transformation and property (4.2), it follows that if $||\Theta|| \leq 1$ and $||\Psi|| \leq 1$ then we have $||\mathcal{D}_1|| \leq 1$ *and,* subsequently, $||\mathcal{M}_1|| \leq 1$. Therefore, the key factor in determining the stability of our method is the way by which the matrices Θ_{ij} and Ψ_{ij} are computed.

To see this, note that, using the series expansion in (3.1) it can be easily shown that the analytical (and thus fully accurate) expression of the matrix Θ_{ij} is given by

$$\Theta_{ij} = e^{T_{ij}} = \begin{pmatrix} \cos(\alpha c_{ij}) & -c_{ij}\sin(\alpha c_{ij}) \\ \frac{1}{c_{ij}}\sin(\alpha c_{ij}) & \cos(\alpha c_{ij}) \end{pmatrix} \tag{4.3}$$

Similar analytical expressions can be also derived for the matrices Θ'_{ij}, Ψ_{ij}, and Ψ'_{ij}. However, the matrix Θ_{ij} as given by (4.3) is not normal for $c_{ij} \# 1$ and an examination of the eigenvalues of the matrix $\Theta^t_{ij}\Theta_{ij}$ can illustrate that $||\Theta_{ij}|| \leq 1$ is only conditionally satisfied. Furthermore, establishing the bound on α to satisfy the stability seems to be very complex. Interestingly, such analytical expressions, though fully accurate, lead to the conditional stability of our method. Note that, alternatively, one can verify the conditional stability by investigating the l-norm or the infinity norm of the matrix Θ_{ij}.

Here, we propose an alternative technique for stable computation of the matrix Θ_{ij} with a rather interesting implication. Let us define two matrices

$$K_{1ij} = \frac{1}{2}(T_{ij} + T^t_{ij}) = \begin{pmatrix} 0 & -\beta_{ij} \\ -\beta_{ij} & 0 \end{pmatrix} \tag{4.4(r)}$$

$$K_{2ij} = \frac{1}{2}(T_{ij} - T^t_{ij}) = \begin{pmatrix} 0 & -\gamma_{ij} \\ \gamma_{ij} & 0 \end{pmatrix} \tag{4.4b}$$

with $\beta_{ij} = \frac{\alpha(c^2_{ij}-1)}{2} > 0$ and $\gamma_{ij} = \frac{\alpha(c^2_{ij}+1)}{2} > 0$. A second-order approximation of the matrix Θ_{ij} is then given by

$$\Theta_{ij} = e^{\frac{K_{2ij}}{2}} e^{K_{1ij}} e^{\frac{K_{2ij}}{2}} \tag{4.5}$$

Using the series expansion in (3.1), the analytical expression of the matrix $e^{\frac{K_{2ij}}{2}}$ is obtained as

$$\chi_{2ij} = e^{\frac{K_{2ij}}{2}} = \begin{pmatrix} \text{Cos}\frac{\gamma_{ij}}{2} & -\sin\frac{\gamma_{ij}}{2} \\ \sin\frac{\gamma_{ij}}{2} & \cos\frac{\gamma_{ij}}{2} \end{pmatrix} \tag{4.6}$$

The matrix χ_{2ij} is orthogonal and hence $||\chi_{2ij}|| = 1$. This also immediately follows from the fact that K_{2ij} *is* a skew-symmetric matrix. ‘J’bus, the stability problem is now reduced to the computation of the matrix $\chi_{1ij}{}^{K_{1ij}}$. To this end, consider a Pade (2,0) approximation of the matrix χ_{1ij}, with a second-order accuracy, given by

$$\chi_{1ij} = (I - K_{1ij} + \frac{1}{2}K^2_{1ij})^{-1} = \begin{pmatrix} \frac{\beta^2_{ij}}{2} + 1 & \beta_{ij} \\ \beta_{ij} & \frac{\beta^2_{ij}}{2} + 1 \end{pmatrix}^{-1} \tag{4.7}$$

The matrix χ_{1ij} is symmetric and hence normal. A necessary and sufficient condition for $\lambda[\chi_{1ij}] \leq 1$ and hence $||\chi_{1ij}|| \leq 1$ is then given by

$$\beta_{ij} \geq 2 \Longrightarrow \alpha \geq \frac{4}{c^2_{min} - 1} \tag{4.8}$$

where $c_{min} = \text{Min}\{c_{ij}\}$. Equation (4.8) represents an interesting result since it gives a

$$\alpha' \leq \frac{1}{8c_{max}} \tag{4.9}$$

where $c_{max} = \mathrm{Max}\{c_{ij}\}$. It should be emphasized that the condition in (4.8) does not represents any *practical* limitation for our method. To see this, let $\xi = \frac{c_{max}}{c_{min}}$. We than have

$$\alpha_{min} = \frac{4\xi^2}{c_{max}^2 - \xi^2} \tag{4.10}$$

'1'bus, insofar as $c_{max} >> \xi$, the condition in (4.1 O) dots not have any practical significance. To see this, consider a practical case of Merietta Basin, discussed in [1], for which c_{max}

V. Some Extensions of the Method

A. Surface and Subsurface Boundary Conditions

A proper treatment of the surface and subsurface boundary conditions is the critical factor in determining the numerical correctness of any method for solving the AWE. Let us first consider the surface boundary conditions.

For many applications, it is more appropriate to use a Neumann type condition, rather than the Dirichlet condition in (2.3), for the surface boundary of the computational model. Such a boundary condition allows to place the source on the surface and eliminates the ghost signal [1]. The Neumann boundary condition can be included by setting (Fig. 1)

$$u_x(x,y,t) = 0 \qquad x, y \epsilon \Omega'_1 \text{ and } 0 \leq t \leq T \tag{5.1}$$

This can be incorporated in our method in a very simple fashion since, according to our notation, (5.1) implies that

$$p(x,y,t) = 0 \qquad x, y \epsilon \Omega'_1 \text{ and } 0 \leq t \leq T \tag{5.2}$$

or, simply $p_{0j}^{(n)} = 0$, for $j = 1$ to N and $m = 1$ to M, which can be easily incorporated in our method without any algorithmic modification.

A more critical issue is the appropriate treatment of the boundary conditions for the subsurface boundaries of the computational model to eliminate the unwanted reflections. A number of approximate algorithm have been proposed for incorporation of absorbing (or, nonreflecting) boundary conditions [9-13]. Reynolds' algorithm [12] seems to be the simplest and most widely used which, using our notation, is given by (Fig. 1)

$$u_x + \frac{1}{c}u_t = 0 \text{ or } p + \frac{1}{c}q = 0 \qquad x, y \epsilon \Omega'_3 \text{ and } 0 \leq t \leq T \tag{5.3a}$$

$$u_y + \frac{1}{c}u_t = 0 \text{ or } r + \frac{1}{c}q = 0 \qquad x, y \epsilon \Omega'_2 \text{ and } 0 \leq t \leq T \tag{5.3b}$$

$$u_y - \frac{1}{c}u_t = 0 \text{ or } r - \frac{1}{c}q = 0 \qquad x, y \epsilon \Omega'_4 \text{ and } 0 \leq t \leq T \tag{5.3c}$$

As shown above, the structure of our method, resulting from the transformation of the AWE into the first-order hyperbolic equations, allows a straightforward handling of the surface boundary conditions. However, this structure does not seem to be appropriate for inclusion of the subsurafce boundary conditions given by (5.3). As suggested by Givoli [13], one of the goals in devising a new, or selecting an existing, absorbing boundary condition is *the* compatibility *of the boundary conditions with the numerical scheme used inside the* computational *domain,* Here, it is worthy to compare our method with other methods on the basis of the compatibility with the conditions in (5.3).

The conventional (explicit or implicit) finite-difference methods arc based on a simultaneous discretization in both time and space. That is, finite- difference schemes are used for calculation of both temporal and spatial derivatives. As a result, both (2.1)- which is a second-order hyperbolic equation- and (5.3)- which represents a set of first-order hyperbolic equations- arc transformed into a same matrix- vector form which allows their integration in a straightforward fashion.

The boundary conditions in (5.3) is also incompatible with our method. However, unlike the Fourier methods, this incompatibility does not arise from the choice of operator for spatial discretization but rather from the technique employed for temporal discretization. Although (2.5)-(2.6) represent first- order hyperbolic equations, they are derived from the sccotld-order equation in (2.1) and thus there seems to be no solution for a direct incorporation of (5.3) into (2.5)-(2.6). An obvious alternative for handling the absorbing boundary conditions in our method is then to usc the same boundary strip technique as for the Fourier methods.

However, an interesting question is whether there are other absorbing boundary conditions more compatible with our method. Our above discussion clearly suggests that the second-ordct boundary conditions such as the one discussed in [1 O] arc compatible with our method. For example, on the boundary Ω_3' such boundary condition is given by

$$u_{tt} + \frac{1}{c}u_{tx} - \frac{c}{2}u_{yy} = 0 \tag{5.4}$$

which, using our notation, it can be written as

$$q_t + \frac{1}{c}q_x - \frac{c}{2}r_y = 0 \tag{5.5}$$

Equations (2.5)-(2.6) and (5.5) can now be combined and written in a compact form as (2.9) with slightly different matrices $A\bar{\delta}_x$ and $B\bar{\delta}_y$. However, further analysis is needed for an efficient incorporation of this second-order boundary condition. More precisely, splitting techniques need to be devised which preserve both the stability and efficiency of the method.

Another absorbing boundary condition highly compatible with our method is the *Damping Technique* suggested in [1 9]. This technique is similar to [18] in the sense that it also include boundary strips but differs from [18] by writing (2.1) as

$$u_{tt} = c^2(u_{xx} + u_{yy}) - vu_t + f \qquad x, y\varepsilon\Omega \text{ and } 0 < t \leq T \tag{5.6}$$

with $v(x,y) = 0$ for $x,y\varepsilon\Omega$ and $v(x, y) > 0$ for $x,y\varepsilon\tilde{\Omega}$ where $\tilde{\Omega}$ is the boundary strip. The choice of optimal function $v(x, y)$ is further discussed in [19]. Using our notation, (5.6) can be written as

$$q_t = c^2(p_x + r_y) - vq + f \qquad x, y\varepsilon\Omega \text{ and } 0 < t \leq T \tag{5.7}$$

The computation of (5.7) can be pcrforrncd in a same fashion as (2.5) by simply replacing the matrix E_1 in (3.10) with a matrix $\tilde{E}_1$ given by

$$\tilde{E}_1 = \begin{pmatrix} \Upsilon & c^2\bar{S} & 0 \\ \bar{S} & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{5.8}$$

where $\Upsilon = \mathrm{Diag}\{v_{ij}\}$ is a diagonal matrix. Furthermore, $v_{ij} \neq 0$ only for grid points belonging to the boundary strips. A second-order approximation of the exponential of matrix $\tilde{E}_1$ is given by

$$e^{\tilde{E}_1} = e^{\frac{\Upsilon}{2}} e^{E_1} e^{\frac{\Upsilon}{2}}$$

The computation of the matrix e^{E_1} is performed as before, The matrix $e^{\frac{\Upsilon}{2}}$ is diagonal with diagonal elements equal to or smaller than 1 which preserves the stability of our method.

In addition to its high compatibility with our method and preserving both the stability and optimality of the computational cost, another advantage of this technique is that it also allows t i me-parallel computation of our method (see below).

B. Higher Order Spatial *Accuracy*

Our method can be also extended to include higher order spatial accuracy. Following Fornberg [17], the operator for a spatial discretization with an accuracy of order $2v$ is a skew-sylnn]etric. Toeplitz matrix given by

$$S_{2v} = \text{Toeplitz}[0, \ldots, 0, s_{-v}, s_{-v+1}, \ldots, s_{-1}, 0, s_{+1}, \ldots, s_{v}, 0, \ldots, 0] \varepsilon \Re^{N \times N}$$

where

$$s_i = \begin{cases} \frac{2(v!)^2(-1)^{i+1}}{i(v+i)!(v-i)!} & \text{for } |i| \leq v \text{ and } i \neq 0 \\ 0 & \text{for } i > v \text{ or } i = 0 \end{cases}$$

Note that, regardless of the accuracy order of spatial discretization, the computation of the method is given by (3.14). Also, the matrices Ml, D_1, M_2, and D_2 have the same structure as before but the matrices C'_1 and G'_1 are now different. For illustrative purposes, let us consider a fourth-order accurate spatial discretization. In this case, the matrix S_4 is given by

$$S4 = \text{Toeplitz}[0, \ldots, 0, \frac{1}{6}, -\frac{4}{3}, 0, \frac{4}{3}, -\frac{1}{6}, 0, \ldots, 0]$$

The matrix G_1 is now given by

$$G_1 = \begin{pmatrix} 0 & c^2 \bar{S}_4 \\ \bar{S}_4 & 0 \end{pmatrix}$$

where $\bar{S}_4 = \text{Diag}\{S_4, S_4, \cdot \ldots S_4\} \varepsilon \Re^{N^2 \times N^2}$. An appropriate splitting of S_4 is given by

$$S4 = S_{41} + S_{42} + S_{43} -\} \ S44$$

where

$$S_{41} = \begin{vmatrix} 0 & 0 & 0 & \ldots & 0 \\ 0 & 0 & 0 & \ldots & 0 \\ \frac{1}{6} & 0 & 0 & \ldots & 0 \\ 0 & \frac{1}{6} & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \ldots & \frac{1}{6} & 0 \end{vmatrix}, \quad S_{42} = \left| \qquad \qquad \right|,$$

$$S_{43} = \begin{pmatrix} 0 & \frac{4}{3} & 0 & \ldots & 0 \\ 0 & 0 & \frac{4}{3} & \ldots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \ldots & 0 & \frac{4}{3} \end{pmatrix}, \quad S_{44} = \left| \qquad \qquad \right|$$

$$G_{11} = \begin{pmatrix} 0 & c^2 \bar{S}_{41} \\ \bar{S}_{44} & 0 \end{pmatrix}, G_{12} = \begin{pmatrix} 0 & c^2 \bar{S}_{42} \\ \bar{S}_{43} & 0 \end{pmatrix}, G_{13} = \begin{pmatrix} 0 & c^2 \bar{S}_{43} \\ \bar{S}_{42} & 0 \end{pmatrix}, G_{14} = \begin{pmatrix} 0 & c^2 \bar{S}_{44} \\ \bar{S}_{41} & 0 \end{pmatrix}$$

The matrix $D_1 = e^{\alpha G_1}$ can be then computed by a repeated application of (3.4) as

$$D_1 = e^{\frac{1}{2}\alpha G_{11}} e^{\frac{1}{2}\alpha G_{12}} e^{\frac{1}{2}\alpha G_{13}} e^{\alpha G_{14}} e^{\frac{1}{2}\alpha G_{13}} e^{\frac{1}{2}\alpha G_{12}} e^{\frac{1}{2}\alpha G_{11}}$$

The ordering in application of (3.4) is arbitrary and thus several forms for computing the matrix D_1 can be derived. However, all these forms are equal in terms of accuracy, stability, and computational cost. The matrices Gil, G$_{r2}$, G$_{13}$, and G$_{r4}$ have a structure similar to those in (3.18). Therefore, using suitable permutations, they can be reduced to block diagonal matrices similar to that in (3.21), wherein the 2 by 2 submatrices on their main diagonals also have a structure similar to that in (3.22). This allows a *fast* and *stable* computation of exponential of these matrices by using the procedure described in the appendix.

Higher order spatial discretization can be treated in a similar fashion by exploiting the skew-symmetric and Toeplitz structure of the matrix S_{2v}. In general, for an order $2v$ spatial accuracy, the matrix S_{2v} is splitted into $2v$ matrices. l'bus, the matrix D_1 is obtained as a product of $2v$ - 1 sparse matrices. It then follows that, for a spatial accuracy of order 2v, the implementation of (3.14) involves a number of 3($2v$ -- 1) sparse matrix-vector multiplications wherein the matrices have a block diagonal structure similar to the matrix O in (3.24) with 2 by 2 submatrices on their main diagonals. Therefore, the computational cost of the method linearly increases with the order of spatial accuracy.

C. Higher Order Temporal Accuracy

Our method can also be extended to include a fourth-order temporal accuracy. Here, we briefly discuss such an extension to illustrate the possibility of achieving higher order temporal accuracy.

Let us again consider a matrix θ with a splitting as $\theta = \theta_1 + \theta_2$ with noncommuting matrices θ_1 and θ_2. Following De Raedt [6],afourth-order approximation of matrix e^{θ} is given by

$$e^{\theta\Delta t} = e^{\frac{1}{2}\theta_1\Delta t} e^{\frac{1}{2}\theta_2\Delta t} e^{L\,\Delta t^3} e^{\frac{1}{2}\theta_2\Delta t} e^{\frac{1}{2}\theta_1\Delta t} + O((\Delta t)^5) \tag{5.13}$$

with $L = \frac{1}{24}[\theta_1 + 2\theta_2,$ [0$_1$, 02]] and $[\theta_1, \theta_2] =$ 0$_1$6'$_2$--0101, which represents the commutator operator. A fourth-order temporal approximation of (2.12) is then given by

$$Q^{(m)} = e^{\frac{1}{2}A\bar{\delta}_x\Delta t} e^{\frac{1}{2}B\bar{\delta}_y\Delta t} e^{C(\Delta t)^3} e^{\frac{1}{2}A\bar{\delta}_x\Delta t} e^{\frac{1}{2}B\bar{\delta}_y\Delta t} Q^{(m-1)} + \frac{1}{2}(F^{(m)} + F^{(m-1)})\Delta t \qquad m = 1 \text{ to } M \tag{5.14}$$

where

$$C = \frac{1}{24}[A\bar{\delta}_x + 2B\bar{\delta}_y, [A\bar{\delta}_x, B\bar{\delta}_y]] = \frac{1}{24}\begin{pmatrix} 0 & -2c^2\delta_y^2c^2\delta_x & c^2\delta_x^2c^2\delta_y \\ -2\delta_x c^2\delta_y^2 & 0 & 0 \\ \delta_y c^2\delta_x^2 & 0 & 0 \end{pmatrix} \tag{5.15}$$

Note that, this splitting allows a fourth-order temporal accuracy while involving only one time level, i.e., only $Q^{(m-1)}$.

The key purpose of this derivation is to illustrate that the resulting matrix C has a sparse structure similar to the matrix E, given by (2.11). Thus, similar techniques can be used for splitting matrix C and fast computation of the term $e^{C(\Delta t)^3}$. However, in order to preserve the fourth-order accuracy, further analysis is needed to derive stable technique for computing the exponential of the resulting two by two matrices (sec $] V. A). To this end, the structure of matrix C is a key incentive for such further analysis.

VI. Techniques for Efficient Space and Time Parallel Computation

The computational complexity of (3.14) is a function of both space (i.e., number of the spatial grid points) and time (i.e., number of the time steps). In this sense, tile computation is performed in two dimensions: space and time. Thus, onc can consider the exploitation of parallelism in one (i.e., either space or time) or in both dimensions (i.e., both space and time). Most conventional approaches for parallel corn put ation of (explicit or implicit) marching-in-time methods are space-parallel, that is, they attempt to exploit parallelism only in computation of each time step. In the following, we first discuss some techniques for a more efficient space-parallel computation of our method. We then discuss a technique that allows the computation of our method to be partially parallelized in time.

A. Space Parallel Computation

At first glance, the sequence of matrix-vector multiplications for implementation of (3.14) seems to be readily amenable to highly efficient parallel computation. In fact, given the block diagonal structure of matrices Θ, Θ', Ψ, and Ψ', their multiplication by any vector can be decomposed into a set of a large number of submatrix-vector multiplications, involving 2 by 2 submatrices, which can be performed in a fully decoupled and parallel fashion. However, such an implementation would require a massive amount of data communication for performing various permutations of vectors which result from matrix-vector multiplications involving the permutation matrices P_{11}, P_{11}^t, P_{12}, and P_{12}^t. in particular, on massively parallel MIMD architectures for which the communication latency is much greater than the cost of floating-point operations, the communication cost of these permutations could be much greater than the computation cost of the algorithm, thus severely degrading the overall performance of such a parallel computation.

An obvious but partial solution to this problem is to reduce the number of permutations by combining the successive permutations in (3.30) and (3.32). That is, by forming a new permutation matrix as $\Pi = P_{11}P_{12}$. Using Π and its transpose, the number of permutations in (3.30) and (3.32) can be reduced from 12 to 8.

A more efficient technique for improving the overall performance of ou r algorithm is based on a tradeoff between computation and communication cost as follows. Let us define the matrices:

$$\tilde{\Theta} = P_{11}^t \Theta P_{11} \text{ and } \tilde{\Psi} = P_{12}^t \Psi P_{12} \tag{5.15}$$

It can be shown that the matrices $\tilde{\Theta}$ and $\tilde{\Psi}$ have a sparse arrd block structure given by

$$\tilde{\Theta} = \begin{pmatrix} \tilde{\Theta}_1 & \tilde{\Theta}_2 \\ \tilde{\Theta}_3 & \tilde{\Theta}_4 \end{pmatrix} \text{ and } \tilde{\Psi} = \begin{pmatrix} \tilde{\Psi}_1 & \tilde{\Psi}_2 \\ \tilde{\Psi}_3 & \tilde{\Psi}_4 \end{pmatrix} \tag{5.16}$$

where the submatrices $\tilde{\Theta}_1, \tilde{\Theta}_4, \tilde{\Psi}_1$, and $\tilde{\Psi}_4$ are diagonal; the submatrices $\tilde{\Theta}_2$ and $\tilde{\Psi}_3$ have nonzero elements only on their upper diagonal; the submatrices $\tilde{\Theta}_3$ and $\tilde{\Psi}_2$ have nonzero elements only on their lower diagonal. Taking advantage of the regular and sparse structure of the matrices $\tilde{\Theta}$ and $\tilde{\Psi}$, it is a rather straightforward task to exploit parallelism in their multiplication by a vector. This, for example, can be achieved by using some of the well known techniques for exploitation of parallelism in matrix-vector multiplications arising in conventional explicit methods. However, even using the best of these techniques, it is very unlikely that a speedup comparable to that by using the matrices Θ and Ψ can be achieved in the computation. The key point, nevertheless, is the fact that exploiting parallelism in multiplication of a vector by the matrices $\tilde{\Theta}$ and $\tilde{\Psi}$ involves a significantly leas amount of data communication than performing the above-mentioned permutations. Therefore, one can expect a much better overall performance as a result of this tradeoff.

We can also introduce additional higher level parallelism in the computation by an appropriate algorithmic modification as follows. An alternative second-order approximation of the matrix $e^{\theta\Delta t}$ is given by

$$e^{\theta\Delta t} = \frac{1}{2}(e^{\theta_1\Delta t}e^{\theta_2\Delta t} + e^{\theta_2\Delta t}e^{\theta_1\Delta t}) + O((\Delta t)^3) \tag{5.17}$$

from which, a second-order, stable, temporal approximation of (2.12) is obtained as

$$Q^{(m)} = \frac{1}{2}(e^{A\bar{\delta}_x\Delta t}e^{B\bar{\delta}_y\Delta t} + e^{B\bar{\delta}_y\Delta t}e^{A\bar{\delta}_x\Delta t})Q^{(m-1)} + \frac{1}{2}(F^{(m)} + F^{(m-1)})\Delta t \qquad m = 1 \text{ to } M \tag{5.18}$$

Note that the matrix $e^{A\bar{\delta}_x\Delta t}$ can be obtained from $e^{A\bar{\delta}_x\Delta t}$ by simply replacing α with 2α, i.e., by replacing α with 2α in matrix Θ given by (4.5). Defining

$$\hat{\mathcal{M}}_1 = e^{A\bar{\delta}_x\Delta t}e^{B\bar{\delta}_y\Delta t} \text{ and } \hat{\mathcal{M}}_2 = e^{B\bar{\delta}_x\Delta t}e^{A\bar{\delta}_y\Delta t}$$

the computation of (5.18) is then performed according to following steps:

$$Q_1^{(m)} = \hat{\mathcal{M}}_1 Q^{(m-1)} \tag{5.19a}$$

$$Q_2^{(m)} = \hat{\mathcal{M}}_2 Q^{(m-1)} \tag{5.19b}$$

$$Q^{(m)} = Q_1^{(m)} + Q_2^{(m)} + \frac{1}{2}(F^{(m)} + F^{(m-1)})\Delta t \tag{5.19c}$$

The unconditional stability of (5.18) follows from the fact that, from the discussion in §IV.A, we have $||\hat{\mathcal{M}}_1|| \leq 1$ and $||\hat{\mathcal{M}}_2|| \leq 1$. Using *the* matrix norm property in

$$\tilde{Q}^{(m)} = e^{-\frac{A}{2}\bar{\delta}_x \Delta t} Q^{(m)} \quad \text{and} \quad \tilde{F}^{(m)} = \frac{1}{2} e^{-\frac{A}{2}\bar{\delta}_x \Delta t}(F^{(m)} + F^{(m-1)})\Delta t \qquad m = 1 \text{ to } M \tag{5.22}$$

$$\tilde{Q}^{(m)} = e^{B\bar{\delta}_y \Delta t} e^{A\bar{\delta}_x \Delta t} \tilde{Q}^{(m-1)} + \tilde{F}^{(m)} \qquad m = 1 \text{ to } M$$

The implementation of (5.23) involves a less amount of computation and communication than (3.5). In fact, its cost is the same as for the first-order approximation given in (3.3). The temporal parallelism in the computation results from the fact the computation of vectors $\tilde{F}^{(m)}$ in (5.22) can be performed fully in parallel for $m = 1$ to M. Also, the vectors $Q^{(m)}$ can be computed from

$$Q^{(m)} = e^{\frac{A}{2}\bar{\delta}_x \Delta t}\tilde{Q}^{(m)}$$

fully in parallel for $m = 1$ to M. Note that the analytical expression can be used for computation of the matrix $e^{\frac{A}{2}\delta_x \Delta t}$ and its inverse $e^{-\frac{A}{2}\bar{\delta}_x \Delta t}$ since, here, stability is not an issue.

VII. Conclusion

In this paper we presented a novel method for solution of the AWE. To our knowledge, this is the first reported method that achieves the computational efficiency of the explicit methods and unconditional stability of the implicit methods. We also discussed several numerical aspects of the method. It should be mentioned, however, that further analysis and numerical experimentation are needed to build a same level of confidence as for more well established methods. Nevertheless, we believe that our results clearly point to a new direction for developing more efficient algorithms for massively parallel solution of the AWE.

Acknowledgment

The research of first author was performed at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration (NASA). The authors would like to thank Dr. F. Aminzadeh from UNOCAL Co. for support and encouragement in initiating this research work. Thanks are also due to Dr. J. Stefani from Chevron Petroleum Technology Co. and Dr. R.E. Scheid from JPL for insightful discussions and suggestions.

References

1. I.R. Mufti, "Large-scale three-dimensional seismic models and their interpretive significance," *Geophysics*, Vol. 55, pp. 1166-1182, 1990.

2. M.A. Dablain, "The application of high-order differencing to the scalar wave equation," *Geophysics*, Vol. 51(1), pp. 54-66, 1986.

3. W.F. Ames, *Numerical methods for Partial Differential Equations*, 2nd Edition, Academic Press, 1977.

4. C. Moler and C. Van Loan, "Nineteen dubious ways to compute the exponential of a matrix," *SIAM Review*, Vol. 20(4), pp. 801- 836, 1978.

5. [illegible]

6. H. De Raedt, "Product formula algorithms for solving the time dependent Schrodinger equation," *Computer Physics Reports*, Vol. 7, pp. 1-72, 1987.

7. C.B. Vreugdenhil, "Accuracy of product-formula algorithms," *J. Comput. Phys.*, Vol. 97, pp. 337-351, 1991.

8. J.L. Richardson, R.C. Ferrell, and L.N. Long, "Unconditionally stable explicit algorithms for nonlinear fluid dynamics problems," *J. Comput. Phys.*, Vol. 104, pp. 69-74, 1993.

9. B. Engquist and A. Majda, "Absorbing boundary conditions for the numerical simulation of waves," *Math. Comp.*, Vol. 31(5), pp. 629-651, 1977.

10. R.W. Clayton and B. Engquist,"Absorbing boundary conditions for acoustic and elastic wave equations," *Bulletin of the Seismological Society of America*, Vol. 67(6), pp. 1529-1540, 1977.

11. R. W. Clayton and B. Engquist,"Absorbing boundary conditions for wave-equation migration," *Geophysics*, Vol. 45(5), pp. 895-904, 1980.

12. A.C. Reynolds,"Boundary conditions for the numerical simulation of wave propagation problems," *Geophysics*, Vol. 43, pp. 1099-1110, 1978.

13. D. Givoli,"Non-reflecting boundary conditions," *J. Comput. Phys.*, Vol. 94, pp. 1-29, 1991.

14. D. Kosloff and E. Baysal,"Forward modeling by a Fourier method," *Geophysics*, Vol. 47(10), pp. 1402-1412, 1982.

15. J.Wen, G. McMechan, and M. Booth,"Three-dimensional modeling and migration of seismic data using Fourier transform," *Geophysics*, Vol. 53(9), pp. 1194-1201, 1988.

16. M. Reshef, D. Kosloff, M. Edwards, and C. Hsiung,"Three-dimensional acoustic modeling by the Fourier method," *Geophysics*, Vol. 53(9), pp. 1175-1183, 1988.

17. B. Fornberg,"The pseudospectral method: comparison with finite differences for the elastic wave equation," *Geophysics*, Vol. 52(4), pp. 483-501, 1987.

18. C. Cerjan, D. Kosloff, R. Kosloff, and M. Reshef,"A nonreflecting boundary condition for discrete acoustic and elastic wave equations," *Geophysics*, Vol. 50(4), pp. 705-708, 1985.

19. J. Sochacki, R. Kubichek, J. George, W.R. Fletcher, and S. Smithson," Absorbing boundary conditions and surface waves," *Geophysics*, Vol. 52(1), pp. 60-71, 1987.

20. [redacted]

21. A. Fijany, M. Jensen, Y. Rahmat-Samii, and J. Barhen," A massively parallel computational strategy for FDTD: time and space parallelism applied to electromagnetics problems," To appear in *IEEE Trans. Antenna & Propagation*, 1995.

22. A. Fijany, J. Barhen, and N. Toomarian" A computational strategy for exploitation of massive temporal parallelism in solution of time-dependent PDEs," *Proc. Toward Teraflop Computing and New Grand Challenge Applications Conf.*, Feb. 1994.

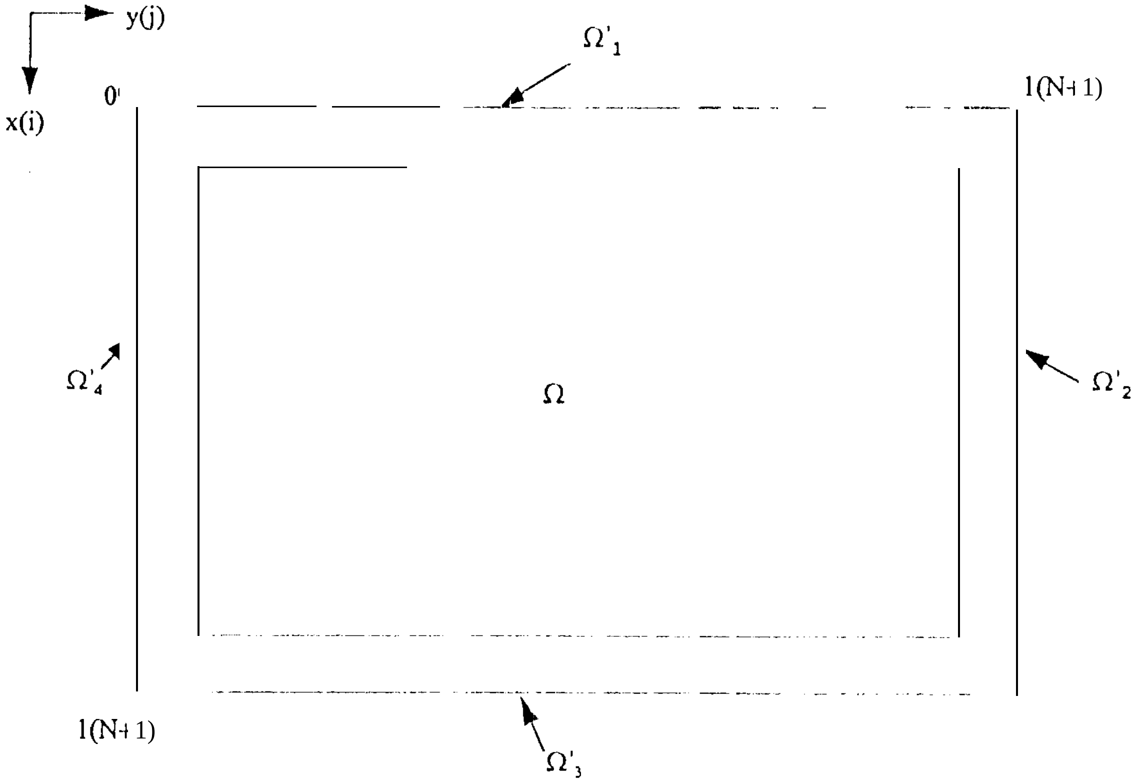

Figure 1: Computational Mesh and Broundaries